(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
22 May 2003 (22.05.2003)

PCT

(10) International Publication Number
WO 03/043342 A1

(51) International Patent Classification[7]: H04N 7/26, 7/36, 7/50

(21) International Application Number: PCT/FI02/00894

(22) International Filing Date:
12 November 2002 (12.11.2002)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
20012203 13 November 2001 (13.11.2001) FI

(71) Applicant *(for all designated States except US)*: **HANTRO PRODUCTS OY** [FI/FI]; Nahkatehtaankatu 2, FIN-90100 Oulu (FI).

(72) Inventor; and

(75) Inventor/Applicant *(for US only)*: **SUVANTO, Markus** [FI/FI]; Tervakukkatie 26 B 14, FIN-90580 Oulu (FI).

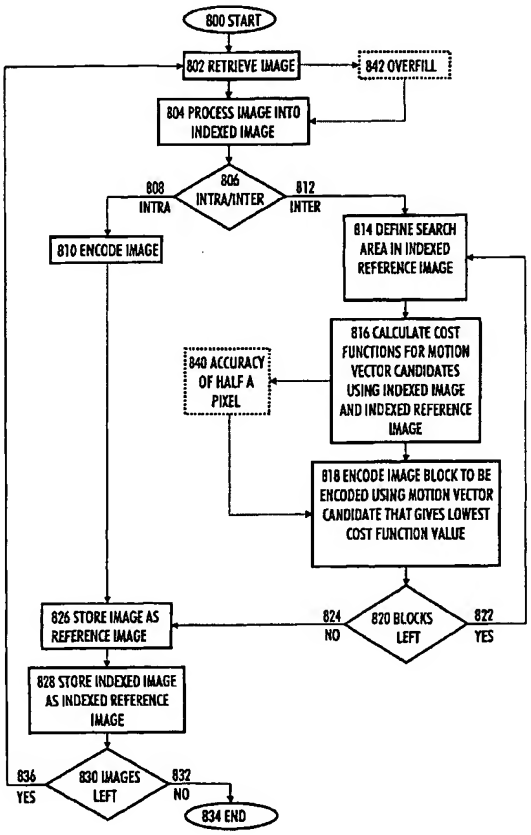
(74) Agent: **KOLSTER OY AB**; Iso Roobertinkatu 23, P.O. Box 148, FIN-00121 Helsinki (FI).

(81) Designated States *(national)*: AE, AG, AL, AM, AT (utility model), AT, AU, AZ, BA, BB, BG, BR, BY, BZ, CA, CH, CN, CO, CR, CU, CZ (utility model), CZ, DE (utility model), DE, DK (utility model), DK, DM, DZ, EC, EE (utility model), EE, ES, FI (utility model), FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MA, MD, MG, MK, MN, MW, MX, MZ, NO, NZ, OM, PH, PL, PT, RO, RU, SC, SD, SE, SG, SI, SK (utility model), SK, SL, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, YU, ZA, ZM, ZW.

(84) Designated States *(regional)*: ARIPO patent (GH, GM, KE, LS, MW, MZ, SD, SL, SZ, TZ, UG, ZM, ZW),

[Continued on next page]

(54) Title: METHOD, APPARATUS AND COMPUTER FOR ENCODING SUCCESSIVE IMAGES



(57) Abstract: The invention relates to a method, an apparatus and a computer program for encoding successive images. The method comprises encoding (818) an image block to be encoded using the motion vector candidate that gives the lowest cost function value. Before encoding, the image is processed (804) into an indexed image and the reference image is indexed into an indexed reference image so that the image and the reference image are divided into parts referred to with indexes and a number is formed from the values of pixels in each part to describe pixel values in a given part; defining (814) a search area in the indexed reference image where the block to be encoded in the indexed image is searched for; and calculating (816) a cost function for each motion vector candidate using the indexed image and the indexed reference image.

WO 03/043342 A1

Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE, SK, TR), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

Declaration under Rule 4.17:

— *as to applicant's entitlement to apply for and be granted a patent (Rule 4.17(ii)) for the following designations AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, BZ, CA, CH, CN, CO, CR, CU, CZ, DE, DK, DM, DZ, EC, EE, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MA, MD, MG, MK, MN, MW, MX, MZ, NO, NZ, OM, PH, PL, PT, RO, RU, SC, SD, SE, SG, SI, SK, SL, TJ, TM, TN, TR, TT, TZ, UA, UG, UZ, VC, VN, YU, ZA, ZM, ZW, ARIPO patent (GH, GM, KE, LS, MW, MZ, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE, SK, TR), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG)*

Published:

— *with international search report*

For two-letter codes and other abbreviations, refer to the "Guidance Notes on Codes and Abbreviations" appearing at the beginning of each regular issue of the PCT Gazette.

METHOD, APPARATUS AND COMPUTER FOR ENCODING SUCCESSIVE IMAGES

FIELD

[0001] The invention relates to a method, an apparatus and a computer for encoding successive images.

BACKGROUND

[0002] Encoding of successive images, e.g. video, is used to reduce the amount of data so that data can be stored more efficiently in a memory means or transmitted using a telecommunications connection. An example of a video encoding standard is MPEG-4 (Moving Images Expert Group). There are different image sizes in use, e.g. the cif size is 352 x 288 pixels and the qcif size 176 x 144 pixels.

[0003] A single image is typically divided into blocks containing information on luminance, colour and location. The data included in the blocks are compressed blockwise by a desired encoding method. The compression is based on deletion of less significant data. Compression methods are mainly divided into three classes: spectral redundancy reduction, spatial redundancy reduction and temporal redundancy reduction. Typically various combinations of these methods are used in compression.

[0004] For example, a YUV colour model is used to reduce spectral redundancy. The YUV model utilizes the fact that the human eye is more sensitive to variations in luminance, i.e. light, than to variations in chrominance, i.e. colour. The YUV model includes one luminance component (Y) and two chrominance components (U and V, or C_b and C_r). For example, a luminance block in accordance with the H.263 video encoding standard is 16 x 16 pixels and both chrominance blocks, which cover the same area as the luminance block, are 8 x 8 pixels. A combination of one luminance block and two chrominance blocks is called a macro block. Each pixel both in the luminance and in the chrominance block may receive a value from 0 to 255, i.e. eight bits are needed to present one pixel. For example, value 0 of the luminance pixel means black and value 255 white.

[0005] Spatial redundancy is reduced using discrete cosine transform DCT, for instance. In discrete cosine transform, the pixel presentation of a block is transformed into a space frequency presentation. Furthermore, in an image block only the signal frequencies that appear in it have high-amplitude

factors, and the factors of the signals which do not appear in the block are close to zero. In principle, discrete cosine transform is a loss-free transform and interference is caused to the signal only in quantization.

[0006] Temporal redundancy is reduced utilizing the fact that successive images usually resemble each other. Thus, instead of compressing each single image, motion data are generated on the blocks. This is called motion compensation. A reference image stored in the memory earlier is searched for as good previously encoded block as possible for the block to be encoded, the motion between the reference block and the block to be encoded is modelled and calculated motion vectors are transmitted to the receiver. The difference between the block to be encoded and the reference block is expressed as difference data. This kind of coding is known as inter-coding, which means utilization of similarities between images of the same image sequence.

[0007] A search area where a block similar to the one in the image to be encoded is searched for is typically defined in the reference image. The best correspondence is found by calculating a cost function between the pixels between the block in the search area and the block to be encoded, e.g. a sum of absolute differences SAD.

[0008] Motion estimation in the search area can be presented by the formula:

$$MV(x,y) = \min_{-16 \le x,y \le 16} \sum_{i=0}^{15} \sum_{j=0}^{15} \left| f_{i,j} - r_{i+x,j+y} \right| \qquad (1)$$

where MV is the final motion vector, f_{xy} is a pixel of the macro block to be encoded, and r_{xy} is a pixel of the reference image in the search area.

[0009] In prior art solutions, one has used full search, i.e. all possible or nearly all possible motion vectors have been set as motion vector candidates. A problem associated with the use of full search is that the number of calculations required is large. For example, if the size of the search area is 48 x 48 pixels, the number of feasible motion vectors is 33 x 33 with an accuracy of one pixel, and the size of the luminance block is 16 x 16 pixels, 16 x 16 = 256 calculations are needed to calculate one sum of absolute differences, and thus 33 x 33 x 256 = 278 784 calculations are needed to calculate the sums of absolute differences of all possible motion vectors per one macro block. An

image of the qcif size, for example, includes 99 macro blocks, i.e. the number of calculations needed is 99 x 278 784 = 27 599 616.

[0010] One has tried to reduce the number of calculations using less comprehensive search methods instead of the full search, such as Three Step Search, Spiral Search and Hierarchical Motion Estimation, where deterioration of the image quality may cause problems.

BRIEF DESCRIPTION

[0011] An object of the invention is to provide an improved method, an improved apparatus and an improved computer program. One aspect of the invention provides a method of encoding successive images according to claim 1. One aspect of the invention provides an apparatus according to claim 9 for encoding successive images. One aspect of the invention provides an apparatus according to claim 17 for encoding successive images. One aspect of the invention provides a computer program according to claim 25 for encoding successive images. The other preferred embodiments of the invention are disclosed in the dependent claims.

[0012] The invention is based on performing motion estimation using an indexed image and an indexed reference image.

[0013] The solution according to the invention provides nearly the same image quality as the classical full search but with fewer calculations.

LIST OF FIGURES

[0014] The preferred embodiments of the invention will be described by examples with reference to the accompanying drawings, in which

Figure 1 illustrates an apparatus for encoding successive images;

Figure 2 illustrates division of a qcif-sized image into blocks;

Figure 3 illustrates part of an image to be encoded;

Figure 4 illustrates part of a reference image;

Figures 5, 6 and 7 illustrate performance of indexing;

Figure 8 is a flow chart illustrating a method of encoding successive images.

DESCRIPTION OF EMBODIMENTS

[0015] Video encoding is well known to a person skilled in the art from standards and textbooks, e.g. from the following works which are incorporated herein by reference: Vasudev Bhaskaran and Konstantinos Konstanti-

nides: *Image and Video Compressing Standards - Algorithms and Architectures*, Second Edition; and Kluwer Academic Publishers 1997, Chapter 6: *The MPEG video standards,* and *Digital Video Processing*, Prentice Hall Signal Processing Series, Chapter 6: *Block Based Methods*.

[0016] The successive images to be encoded are typically moving images, e.g. video. In the camera, a video image consists of individual successive images. The camera forms a matrix which presents the images as pixels e.g. in the manner described above, where luminance and chrominance have separate matrixes. The data flow that presents the image as pixels is supplied to an encoder. It is also feasible to build a device where data flow is transmitted to the encoder along a data transmission connection, for example, or from the memory means of a computer. In that case the purpose is to compress an uncompressed video image with an encoder for forwarding or storage. The compressed video image formed by the encoder is transmitted along a channel to a decoder. In principle, the decoder performs the same functions as the encoder when it forms an image but inversely. The channel may be, for example, a fixed or a wireless data transmission connection. The channel can also be interpreted as a transmission path which is used for storing the video image in a memory means, e.g. on a laser disc, and by means of which the video image is read from the memory means and processed in the decoder. Encoding of other kind can also be performed on the compressed video image to be transmitted on the channel, e.g. channel coding by a channel coder. Channel coding is decoded by a channel decoder. The encoder and decoder may be arranged in different devices, such as computers, subscriber terminals of different radio systems, e.g. mobile stations, or in other devices where video is to be processed. The encoder and decoder can also be connected to the same device, which can be called a video codec. The structure of a device for encoding successive images, i.e. an encoder, will be described with reference to Figure 1.

[0017] Figure 1 describes the function of the encoder on a theoretical level. In practice, the structure of the encoder will be more complicated since a person skilled in the art adds necessary prior art features to it, such as timing and blockwise processing of images. Successive images 130 are supplied to a frame buffer 102 for temporary storage. A single image 132 is supplied from the frame buffer 102 to block 104, where the desired coding mode is selected. The function of the device is controlled by a control part 100, which

selects the desired coding mode and informs block 104 and block 120 of the selected coding mode 156, 158, for instance. The coding mode may be intra-coding or inter-coding. Motion compensation is not performed on an intra-coded image whereas an inter-coded image is compensated for motion. Usually the first image is intra-coded and the following images are inter-coded. Intra-images can also be transmitted after the first image if, for example, sufficiently good motion vectors are not found for the image to be encoded.

[0018] In the following, the function of the apparatus will be described in a situation where intra-coding has been selected in block 104.

[0019] Block 104 receives only the image 132 arriving from the frame buffer 102 as input for the intra-image. The image 132 obtained from the frame buffer 102 is supplied as such 134 to a discrete cosine transform block 106 where the discrete cosine transform described at the beginning is performed.

[0020] The image 136 on which discrete cosine transform has been performed is supplied to a quantization block 108, where quantization is performed, i.e. in principle each element of the image on which discrete cosine transform has been performed is divided by a constant and the result of the division is rounded to an integer. This constant may vary between different macro blocks. A quantization parameter, from which the divisors are calculated, is typically between 1 and 31. The more zeroes the block includes, the better it can be packed since zeroes are not transmitted to the channel.

[0021] Then the quantisized image 138 on which discrete cosine transform has been performed is supplied to a variable length coder 110, which outputs the encoded image 140 produced by the device.

[0022] In addition to the variable length coder 110, the quantized image 138 on which discrete cosine transform has been performed is taken from the quantization block 108 to an inverse quantization block 112, which performs inverse quantization on the input quantized image 138 on which discrete cosine transform has been performed, i.e. restores it to image 136 as accurately as possible. Then the image 142 quantized inversely is supplied to an inverse discrete cosine transform block 114, where inverse discrete cosine transform is performed. Since the discrete cosine transform is a loss-free transform and quantization is not, image 144 does not completely correspond to image 134. The purpose of inverse quantization and inverse discrete cosine transform is to produce an image in the encoder which is similar to the one

produced by the decoder corresponding to the encoding device. The 'decoded' image 144 is then supplied to block 124, where the part deleted from the image, i.e. difference data, would be added to it if the image had been inter-coded. Since the image in question is intra-coded, nothing is added to it. This decision is made by block 120, where intra-coding is the pre-selected option, in which case there is nothing in the input of block 120 and thus nothing is included in the output 154 connected to its block 124. After this, the intra-image 146 is stored in the frame buffer 116. Thus a reconstructed image is stored in the frame buffer 116, i.e. the encoded image in the form in which it is after decoding performed in the decoder. There are two frame buffers: the image arriving at the device is stored in the first buffer 102 and the reconstructed 'previous' image is stored in the second buffer 116. The above described how to process an image for which intra-coding had been selected in blocks 104 and 120.

[0023] One can start using motion compensation in the processing of the next image.

[0024] In that case inter-coding is selected in blocks 104 and 120. The image 116 stored in the frame buffer is now a reference image and the image to be encoded is the image 132 to be obtained next from the frame buffer 102. As appears from Figure 1, the next image is supplied to a motion estimation block 118 in addition to block 104. The motion estimation block 118 also receives a reference image 150 from the frame buffer 116. The function of the motion estimation block 118 will be described in greater detail below. At this point it is sufficient to note that the block searches the reference image for blocks corresponding to the blocks in the image to be encoded. Transitions between the blocks are expressed as motion vectors 152, 166, which are supplied both to the variable length coder and to the frame buffer 116.

[0025] The reference image 148 is taken from the frame buffer 116 to block 122. Block 122 subtracts the reference image 148 from the image 132 to be encoded to provide difference data 164, which are supplied from block 104 via the discrete cosine transform block 106 and quantization block 108 to the variable length coder 110.

[0026] The variable length coder 110 encodes the difference data 138 and motion vectors 166, in which case the output 140 of the variable length coder 110 provides an inter-coded image. The variable length coder 110 receives as inputs the quantized difference data 138 on which discrete

cosine transform has been performed and motion vectors 166. The output 140 of the encoder thus provides the inter-coded image with compressed data, which represent the encoded image and the encoded image in relation to the reference image by means of motion vectors and difference data. The motion estimation is carried out using the luminance blocks but the difference data to be encoded are calculated both for the luminance and the chrominance block.

[0027] Inverse quantization is also performed on the difference data 138 of the inter-coded image in the inverse quantization block 112 and inverse discrete cosine transform in the inverse discrete cosine transform block 114. The difference data 144 processed this way are supplied to block 124, where the previous image 154 subtracted in the encoding of the inter-image in question and obtained from the place indicated by the motion vector is added to the difference data. The sum 146 of the difference data and the previous image is supplied from block 124 to the frame buffer 116 to obtain a reconstructed image. The reconstructed image corresponds to the image obtained in the decoder when the encoding of the inter-coded image 140 is decoded. Thus the frame buffer 116 has a reference image ready for encoding of the image 132 received next from the frame buffer 102.

[0028] The control block 100 controls the function of the encoder. In addition to selection of the coding mode, it controls selection 160 of the correct quantization ratio and performance 162 of encoding with a variable length, for instance. The control block 100 may also control other encoder blocks even though this is not illustrated in Figure 1. For example, the function of the motion estimation block 118 is controlled by the control block 100.

[0029] In the following, a method of encoding successive images will be described with reference to the flow chart shown in Figure 8. The encoding is presented expressly in respect of reduction of temporal redundancy and no other methods of redundancy reduction are described here. The method starts in block 800, where the encoder is started. In block 802, the next image is retrieved from the frame memory. The image may be e.g. the qcif-sized image with 176 x 144 luminance pixels illustrated in Figure 2. The image is divided into macro blocks 200 whose luminance parts have the size of 16 x 16 pixels. The macro blocks comprise eleven columns and nine rows. The luminance pixels can be presented by the matrix

$$f_{i,j} = \begin{bmatrix} f_{0,0} & f_{0,1} & f_{0,2} & \cdots & f_{0,175} \\ f_{1,0} & f_{1,1} & f_{1,2} & \cdots & f_{1,175} \\ f_{2,0} & f_{2,1} & f_{2,2} & \cdots & f_{2,175} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ f_{143,0} & f_{143,1} & f_{143,2} & \cdots & f_{143,175} \end{bmatrix}, \quad (2)$$

where i=0,1,2,...,143 and j=0,1,2,...,175.

[0030] In block 804, the image is processed into an indexed image by dividing the image into parts referred to with indexes and forming a number from the values of pixels in each part to describe pixel values in the part concerned.

[0031] In an embodiment, the pixels included in the part form a square because this is advantageous according to the experiments carried out by the applicant. The image size and block size set certain limits on the size of the indexed part used in the method. According to the applicant's experiments, it is advantageous in the case of a qcif-sized image that the part includes 4 x 4 pixels. In that case matrix 2 can be presented as follows:

$$F_{I,J} = \begin{bmatrix} F_{0,0} & F_{0,1} & F_{0,2} & \cdots & F_{0,172} \\ F_{1,0} & F_{1,1} & F_{1,2} & \cdots & F_{1,172} \\ F_{2,0} & F_{2,1} & F_{2,2} & \cdots & F_{2,172} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ F_{140,0} & F_{140,1} & F_{140,2} & \cdots & F_{140,172} \end{bmatrix}, \quad (3)$$

where I=0,1,2,..., 140 and J=0,1,2,...,172.

[0031] In addition to indexing, a number describing pixel values in a given part is formed from the values of pixels in each part. The simplest way to obtain this number is to add together the numerical values of the pixels in the part concerned. The number for each part of matrix 3 is obtained from matrix 2 by the following formula:

$$F_{I,J} = \sum_{i=I}^{I+3} \sum_{j=J}^{J+3} f_{ij} \quad (4)$$

[0032] For example, the number for the part referred to with index $F_{0,0}$ is obtained as follows

$$F_{0,0} = \sum_{i=0}^{3}\sum_{j=0}^{3} f_{ij} \,, \tag{5}$$

the number for the part referred to with index $F_{0,1}$ is obtained as follows

$$F_{0,1} = \sum_{i=0}^{3}\sum_{j=1}^{4} f_{ij} \,, \tag{6}$$

the number for the part referred to with index $F_{1,0}$ is obtained as follows

$$F_{1,0} = \sum_{i=1}^{4}\sum_{j=0}^{3} f_{ij} \,, \tag{7}$$

and the number for the part referred to with index $F_{1,1}$ is obtained as follows

$$F_{1,1} = \sum_{i=1}^{4}\sum_{j=1}^{4} f_{ij} \,. \tag{8}$$

[0033] When matrixes 2 and 3 are compared, it can be noted that the parts referred to with indexes and located close to each other partly overlap. This means that the areas referred to with indexes $F_{0,0}$ and $F_{1,0}$, for example, include twelve same pixels of matrix 2, i.e. pixels $f_{1,0}$, $f_{1,1}$, $f_{1,2}$, $f_{1,3}$, $f_{2,0}$, $f_{2,1}$, $f_{2,2}$, $f_{2,3}$, $f_{3,0}$, $f_{3,1}$, $f_{3,2}$ and $f_{3,3}$. When the numbers for two adjacent parts referred to with indexes are calculated in an embodiment, the number already calculated for the second part is utilized in the calculation of the number for the first part. This principle of sliding calculation can be described as follows. As stated above, the number for the part referred to with index $F_{0,0}$ can be calculated using formula 5. In that case the other numbers are obtained in a sliding manner

$$F_{1,0} = F_{0,0} - \sum_{j=0}^{3} f_{0,j} + \sum_{j=0}^{3} f_{4,j} \,, \text{ ja} \tag{9}$$

$$F_{2,0} = F_{1,0} - \sum_{j=0}^{3} f_{1,j} + \sum_{j=0}^{3} f_{5,j} \tag{10}$$

[0034] The same principle can be applied to the calculation of all numbers, also when one proceeds from the left side of matrix 2 to its right side.

[0035] After the image has been processed into an indexed image in block 804 according to the principles described above, one can move to block 806, where the coding mode to be used, i.e. intra-coding or inter-coding, is selected.

[0036] If the selected coding mode is intra-coding, we proceed in accordance with arrow 808 from block 806 to block 810, where the image is encoded, i.e. discrete cosine transform and quantization are performed on it but no motion estimation. Then we move to block 826, where the intra-coded image is stored as a reference image. After this, the indexed image is stored in block 828 as an indexed reference image referred to with indexes. Furthermore, a number is formed from the values of pixels in each part to describe pixel values in the part concerned. It should be noted that the fact that the image has been processed into an indexed image in block 804 is utilized here. It should also be noted that the operations of blocks 826 and 828 may be optional since the reference image does not necessarily need to be the image that immediately precedes the image to be encoded but it can also be an earlier image.

[0037] From block 828 we move to block 830, where it is checked whether there are images to be encoded left. If there are no images left, we move in accordance with arrow 832 to block 834, where the method ends. If there are images left, we move in accordance with arrow 836 to block 802, where the next image is retrieved from the frame memory.

[0038] If the coding mode selected in block 806 is inter-coding, we move in accordance with arrow 812 from block 806 to block 814, where a search area where the block to be encoded in the indexed image is searched for is defined for the indexed reference image. Inter-coding cannot thus be performed on the first image because it requires at least one reference image. In our example there has to be one intra-coded image, i.e. the operations of blocks 810, 826 and 828 have to have been performed on the reference image in question. In further phases, the reference image may naturally be an image which has been inter-coded earlier.

[0039] Figures 3 and 4 illustrate how a search is performed. Figure 3 illustrates part of an image 300 to be encoded and Figure 4 part of a reference image 400. The parts 300, 400 illustrate the same point of the qcif-sized

image shown in Figure 2. The image 300 to be encoded thus consists of luminance blocks with a size of 16 x 16 pixels. The size of the chrominance blocks is usually 8 x 8 pixels but they are not shown in Figures 3 and 4 because chrominance blocks are not utilized in motion estimation. It should be noted that, for the sake of clarity, Figures 3 and 4 describe the real content of the images without indexing.

[0040] In our example it is assumed that the image includes nothing else but an image which fits exactly in one block 302 and consists of diagonal lines and letter H. A search area 402 is thus defined in the reference image 400, which is the indexed reference image in our example. This area is searched for an image element included in the image to be encoded, i.e. the indexed encoded image in our example. The image element is located in block 302. The search for motion vectors is usually limited to a search area 402 with a size of [-16, 16], in which case the search area 402 consists of nine blocks of 16 x 16 pixels. The nine blocks of the search area 402 are located in the reference image 400 in the manner shown in Figure 4 around the location of the block 302 to be encoded in the image to be encoded 300. The size of the search area 402 is thus 48 x 48 pixels. In that case the number of possible motion vectors, i.e. motion vector candidates, is 33 x 33.

[0041] After this, we move to block 816, where a cost function is calculated for each motion vector candidate using the indexed image and the indexed reference image. Figures 5, 6 and 7 illustrate indexing by describing part of the search area 402 of Figure 4 pixel-by-pixel 502. Figure 5 shows which ones of the areas 500 referred to with indexes are needed in the calculation of motion vector candidate [0,0], i.e. the following elements of matrix 3 are used:

$$F = \begin{bmatrix} F_{0,0} & F_{0,4} & F_{0,8} & F_{0,12} \\ F_{4,0} & F_{4,4} & F_{4,8} & F_{4,12} \\ F_{8,0} & F_{8,4} & F_{8,8} & F_{8,12} \\ F_{12,0} & F_{12,4} & F_{12,8} & F_{12,12} \end{bmatrix} \quad (11)$$

[0042] The following elements of matrix 3 are needed to calculate motion vector candidate [1,0] according to Figure 6:

$$F = \begin{bmatrix} F_{0,1} & F_{0,5} & F_{0,9} & F_{0,13} \\ F_{4,1} & F_{4,5} & F_{4,9} & F_{4,13} \\ F_{8,1} & F_{8,5} & F_{8,9} & F_{8,13} \\ F_{12,1} & F_{12,5} & F_{12,9} & F_{12,13} \end{bmatrix} \quad (12)$$

[0043] The numbers needed to calculate all possible motion vector candidates are obtained by the same principle from matrix 3. For example, according to Figure 7, the following elements of matrix are needed for motion vector candidate [4,0]:

$$F = \begin{bmatrix} F_{0,4} & F_{0,8} & F_{0,12} & F_{0,16} \\ F_{4,4} & F_{4,8} & F_{4,12} & F_{4,16} \\ F_{8,4} & F_{8,8} & F_{8,12} & F_{8,16} \\ F_{12,4} & F_{12,8} & F_{12,12} & F_{12,16} \end{bmatrix} \quad (12)$$

[0044] Motion estimation in the search area can be presented by the formula:

$$MV(x,y) = \min_{-16 \le x,y \le 16} \sum_{i=0}^{3} \sum_{j=0}^{3} \left| F_{i*4,j*4} - R_{i*4+x,j*4+y} \right| \quad (13)$$

where MV is the final motion vector, F_{xy} is the number calculated for the index of the number to be encoded, and R_{xy} is the number calculated for the index of the reference image. When formula 13 used in our method is compared with prior art formula 1, it can be noted that motion estimation requires 33 x 33 x 4 x 4 = 17424 calculations in our method, i.e. 16 times fewer calculations than the prior art full search method. Calculation of indexes has not been taken into account here, but they need to be calculated only once for the whole image. In our example, the SAD function (Sum of Absolute Differences) is used as the cost function but it is clear to a person skilled in the art that other cost functions may also be used if the image indexing method can be utilized in their calculation.

[0045] In our example shown in Figures 3 and 4, a block 404 corresponding to the block 302 to be encoded in the (indexed) image 300 to be encoded was found in the (indexed) reference image. Motion of the block 302 to be encoded with respect to the block 404 found in the reference image is ex-

pressed by a motion vector 406. The motion vector may be described as a motion vector of the pixel of the leftmost upper corner in the block 302 to be encoded, for instance. The other pixels of block 302 naturally also move in the direction of the motion vector concerned.

[0046] The origin (0, 0) of the image is usually the pixel in the left upper corner in the image. In video encoding terminology motions are expressed as follows: a motion to the right is positive, a motion to the left is negative, a motion up is negative and a motion down is positive. The motion vector 406 is (12, -4), i.e. the motion is twelve pixels to the right in the direction of the X axis and four pixels up in the direction of the Y axis.

[0047] From block 816 we then move to block 818, where the image block to be encoded is encoded using the motion vector candidate that gives the lowest cost function value. The motion vector candidate thus defines the motion between the image block to be encoded and the candidate block in the search area of the reference image.

[0048] Then we move to block 820, where it is tested whether blocks to be encoded are still left in the image to be encoded. If there are blocks to be encoded left, we move in accordance with arrow 822 to block 814, where the search for the block corresponding to the next block to be encoded starts in the reference image. The loop according to arrow 822 is repeated until the blocks of the image to be encoded have been processed in the desired manner, either all or some of them.

[0049] If there are no more blocks to be encoded left, we move in accordance with arrow 824 first from block 820 to block 826 and then to block 828, where, if desired, the image that was just encoded is stored as a reference image and the indexed and encoded image is stored as an indexed reference image. Then it is checked in block 830 whether there are still images to be encoded left. If there are no images to be encoded left, we move in accordance with arrow to block 834, where the method ends; otherwise the process continues in accordance with arrow 836 to block 802, where the next image to be encoded is searched for.

[0050] In the method described in Figure 8, the processing of the image to be encoded into an indexed image is utilized so that the result of the same processing can also be stored as an indexed reference image. This reduces the number of necessary calculations, but if the use of memory is to be

optimized, the indexed reference image can be calculated from the stored reference image just before it is used.

[0051] After the motion vector candidate giving the lowest cost function value has been found with an accuracy of one pixel in an embodiment, the area around the motion vector candidate in question is searched for the best motion vector candidate with an accuracy of half a pixel. This is illustrated in Figure 8 with block 840. The location of the 16 x 16 block found in motion estimation of half a pixel is still checked with an accuracy of half a pixel. Usually this requires an 18 x 18 matrix so that the pixels can be interpolated, but our method, which is based on indexes, enables the fact that the area to be interpolated is a 6 x 6 matrix which employs indexes. A motion vector of half a pixel is obtained by applying formula 13, in which case the method also needs 16 times fewer calculations that the traditional motion estimation with an accuracy of half a pixel.

[0052] In an embodiment, the best motion vector candidate is searched for with an accuracy of half a pixel as follows:

interpolating values of half a pixel for the indexed candidate block found, which corresponds to the one-pixel motion vector candidate in the reference image, and around the block;

calculating a cost function for each motion vector candidate of half a pixel using the indexed image and interpolated and indexed candidate block:

encoding the image block to be encoded using the motion vector candidate of half a pixel that gives the lowest cost function value.

[0053] An embodiment utilizes the well known fact that existing encoding standards also allow motion vectors pointing outside the image. In the indexed motion estimation described, this is achieved by overfilling the index table so that there are 16 pixels on each edge of the image, i.e. at the top, at the bottom and on the sides, which have been copied there from the outer edges of the actual image area. This can be performed using block 842 of Figure 8. The size of the indexed matrix 3 is the image size minus three, i.e. in our example 173 x 141. When the overfilling is taken into account, the image size is 173+32 x 141+32, or 176+29 x 144+29. Number 29 is naturally obtained by subtracting three from the space required by overfill, i.e. number 32.

[0054] In the following, calculation of an indexed table and overfilling of indexes are described by a pseudo code which employs the syntax of the C programming language.

```
typedef struct {
   u8 *image;     /* Pointer to image to be indexed      */
   u16 pels;      /* Number of columns in image          */
   u16 lines;     /* Number of rows in image             */
   u16 *mv;       /* Pointer to index table              */
} index_s;

void MeIndex(index_s *vop)
{
   u16 i, j;
   u16 pelsMv;
   u16 *tmp  = NULL;
   u16 *tmp1 = NULL;
   u16 *put  = NULL;
   u16 *start= NULL;
   u8 *block = NULL;

   /* Temporary memory of one index column */
   tmp   = (u16 *)malloc(sizeof(u16)*vop->lines);

   /* Number of columns in index table, */
   /* space required by overfill taken into account    */
   pelsMv = vop->pels+29;

   /* Pointer to first index, represents motion vector (0,0) */
   start  = vop->mv+pelsMv*16+16;

   /* Calculate first column of index table */
   block = image;
   put   = start;
   for (j = 0; j < 4; j++) {
      tmp[j] = block[0]+block[1]+block[2]+block[3];
      block += vop->pels;
   }
   put[0] = tmp[0]+tmp[1]+tmp[2]+tmp[3];
   tmp1   = tmp;
```

```
    for (j = 4; j < vop->lines; j++) {
      tmp1[4]    = block[0]+block[1]+block[2]+block[3];
      put[pelsMv] = put[0]+tmp1[4]-tmp1[0];
      block     += vop->pels; tmp1++; put += pelsMv;
    }

    /* Calculate the rest of columns in index table */
    for (i = 1; i < vop->pels-3; i++) {
      block = image+i;
      put   = start+i;
      for (j = 0; j < 4; j++) {
        tmp[j] = tmp[j]+block[3]-block[-1];
        block += vop->pels;
      }
      put[0] = tmp[0]+tmp[1]+tmp[2]+tmp[3];
      tmp1   = tmp;
      for (j = 4; j < vop->lines; j++) {
        tmp1[4]    = tmp1[4]+block[3]-block[-1];
        put[pelsMv] = put[0]+tmp1[4]-tmp1[0];
        block     += vop->pels; tmp1++; put += pelsMv;
      }
    }

    free(tmp);

    /* Overfill index table */
    IndexOverfill(vop);

    return;
  }

  void IndexOverfill(index_s *vop)
  {
    u16 i, j;
    u16 pelsMv;
    u16 *put = NULL;   /* Help variable, pointer to index table */
```

```
u16 *get = NULL;   /* Help variable, pointer to index table */

/* Number of columns in index table, */
/*  Space required by overfill taken into account      */
pelsMv = vop->pels + 29;

/* Overfill middle section of table from top */
put = vop->mv + 16;
get = vop->mv + pelsMv*16 + 16;
for (j = 0; j < 16; j++) {
   for (i = 0; i < vop->pels; i++) {
      put[i] = get[i];
   }
   put += pelsMv;
}

/* Overfill middle section of table from bottom */
put = vop->mv + (vop->lines+13)*pelsMv+16;
get = vop->mv + (vop->lines+12)*pelsMv+16;
for (j = 0; j < 16; j++) {
   for (i = 0; i < vop->pels; i++) {
      put[i] = get[i];
   }
   put += pelsMv;
}

/* Overfill table from left */
put = vop->mv;
get = vop->mv + 16;
for (j = 0; j < vop->lines+29; j++) {
   for (i = 0; i < 16; i++) {
      put[i] = *get;
   }
   put += pelsMv; get += pelsMv;
}
```

```
    /* Overfill table from right */
    put = vop->mv + vop->pels+13;
    get = vop->mv + vop->pels+12;
    for (j = 0; j < vop->lines+29; j++) {
      for (i = 0; i < 16; i++) {
        put[i] = *get;
      }
      put += pelsMv; get += pelsMv;
    }

    return;
  }
```

[0055] The method described can be implemented using the encoder shown in Figure 1, for example. The encoder shown in Figure 1, i.e. the apparatus for encoding successive images, comprises means 110 for encoding the image block to be encoded using the motion vector candidate that gives the lowest cost function value. The motion vector candidate defines the motion between the image block to be encoded and the candidate block in the search area of the reference image. The apparatus further comprises means 118 for

processing the image into an indexed image and the reference image into an indexed reference image so that the image and the reference image are divided into parts referred to with indexes and a number is formed from the values of pixels in each part to describe pixel values in a given part;

defining a search area in the indexed reference image where the block to be encoded in the indexed image is searched for; and

calculating a cost function for each motion vector candidate using the indexed image and the indexed reference image.

[0056] The apparatus can also be configured to encode the image block to be encoded using the motion vector candidate that gives the lowest cost function value. The motion vector candidate defines the motion between the image block to be encoded and a candidate block in the search are of a reference image. In that case the apparatus is also configured to:

process the image into an indexed image and the reference image into an indexed reference image so that the image and the reference image

are divided into parts referred to with indexes and a number is formed from the values of pixels in each part to describe pixel values in a given part;

define a search area in the indexed reference image where the block to be encoded in the indexed image is searched for; and

calculate a cost function for each motion vector candidate using the indexed image and the indexed reference image.

[0057] The encoder blocks shown in Figure 1 can be implemented as one or more application-specific integrated circuits ASIC. Other embodiments are also feasible, such as a circuit built of separate logic components, or a processor with its software. A hybrid of these different embodiments is also feasible. When selecting the method of implementation, a person skilled in the art will consider the requirements set on the size and power consumption of the device, necessary processing capacity, production costs and production volumes, for example. The above-mentioned means can be placed in the encoder blocks described or they can be implemented as new blocks related to the blocks described. For example, the means for processing an image into an indexed image and a reference image into an indexed reference image can be implemented in block 118 or in the frame buffer 102, or using a new block connected to the frame buffer 102. The device can also be configured using the described blocks or new blocks. One embodiment of the encoder is a computer program on a carrier for encoding successive images, comprising computer executable instructions for causing a computer to perform the encoding when the software is run. The carrier can be any means for distributing the software to the customers. The carrier can be a distribution package (containing a diskette, CD-ROM or another computer readable medium for storing the software), a computer memory (for example a programmed memory chip or another memory device connectable to the computer), a telecommunications signal (for example a signal transferred in the Internet and/or in a cellular radio network containing the software in normal or compressed format). The encoder may also be part of a complete video codec.

[0058] Even though the invention was described above with reference to an example according to the accompanying drawings, it is clear that the invention is not limited thereto but it may be modified in various ways within the inventive concept disclosed in the appended claims. Thus the size of images to be processed may differ from the qcif size used in the example. This does not significantly change the implementation of the invention.

CLAIMS

1. A method of encoding successive images, comprising the following steps:

encoding (818) an image block to be encoded using the motion vector candidate that gives the lowest cost function value, the motion vector candidate defining the motion between the image block to be encoded and a candidate block in the search area of a reference image;

characterized by prior to encoding:

processing (804) the image into an indexed image and the reference image into an indexed reference image so that the image and the reference image are divided into parts referred to with indexes and a number is formed from the values of pixels in each part to describe pixel values in a given part;

defining (814) a search area in the reference image where the block to be to be encoded in the indexed image is searched for; and

calculating (816) a cost function for each motion vector candidate using the indexed image and the indexed reference image.

2. A method according to claim 1, **characterized** in that the pixels included in the part form a square.

3. A method according to claim 2, **characterized** in that the part includes 4 x 4 pixels.

4. A method according to any one of the preceding claims, **characterized** in that a SAD function (Sum of Absolute Differences) is used as the cost function.

5. A method according to any one of the preceding claims, **characterized** in that parts located close to each other and referred to with indexes partly overlap.

6. A method according to claim 5, **characterized** in that when numbers are calculated for two adjacent parts referred to with indexes, the number calculated for the second part is utilized in the calculation of the number for the first part.

7. A method according to any one of the preceding claims, **characterized** in that after the motion vector candidate giving the lowest cost function value has been found with an accuracy of one pixel, the best motion

vector candidate is searched for with an accuracy of half a pixel around the motion vector candidate concerned.

8. A method according to claim 7, **characterized** in that the best motion vector candidate with an accuracy of half a pixel is searched for as follows:

interpolating values of half a pixel for the indexed candidate block found, which corresponds to the one-pixel motion vector candidate in the reference image, and around the block;

calculating a cost function for each motion vector candidate of half a pixel using the indexed image and interpolated and indexed candidate block:

encoding the image block to be encoded using the motion vector candidate of half a pixel that gives the lowest cost function value.

9. An apparatus for encoding successive images, comprising:

means (110) for encoding an image block to be encoded using the motion vector candidate that gives the lowest cost function value, the motion vector candidate defining the motion between the image block to be encoded and a candidate block in the search area of a reference image;

characterized in that the apparatus further comprises:

means (118) for processing the image into an indexed image and the reference image into an indexed reference image so that the image and the reference image are divided into parts referred to with indexes and a number is formed from the values of pixels in each part to describe pixel values of a given part;

means (118) for defining a search area in the indexed reference image where the block to be encoded in the indexed image is searched for; and

means (118) for calculating a cost function for each motion vector candidate using the indexed image and the indexed reference image.

10. An apparatus according to claim 9, **characterized** in that the pixels included in the part form a square.

11. An apparatus according to claim 10, **characterized** in that the part includes 4 x 4 pixels.

12. An apparatus according to any one of claims 9 to 11, **characterized** in that an SAD function (Sum of Absolute Differences) is used as the cost function.

13. An apparatus according to any one of claims 9 to 12, **characterized** in that parts located close to each other and referred to with indexes partly overlap.

14. An apparatus according to claim 13, **characterized** in that when numbers are calculated for two adjacent parts referred to with indexes, the number calculated for the second part is utilized in the calculation of a number for the first part.

15. An apparatus according to any one of preceding claims 9 to 14, **characterized** in that after the motion vector candidate giving the lowest cost function value has been found with an accuracy of one pixel, the best motion vector candidate is searched for with an accuracy of half a pixel around the motion vector candidate in question.

16. An apparatus according to claim 15, **characterized** in that the best motion vector candidate with an accuracy of half a pixel is searched for as follows:

interpolating values of half a pixel for the indexed candidate block found, which corresponds to the one-pixel motion vector candidate in the reference image, and around the block;

calculating a cost function for each motion vector candidate of half a pixel using the indexed image and interpolated and indexed candidate block:

encoding the image block to be encoded using the motion vector candidate of half a pixel that gives the lowest cost function value.

17. An apparatus for encoding successive images which is configured to:

encode an image block to be encoded using the motion vector candidate that gives the lowest cost function value, the motion vector candidate defining the motion between the image block to be encoded and a candidate block in the search area of a reference block;

characterized in that the apparatus is further configured to:

process the image into an indexed image and the reference image into an indexed reference image so that the image and the reference image are divided into parts referred to with indexes and a number is formed from the values of pixels in each part to describe pixel values in a given part;

define a search area in the indexed reference image where the block to be encoded in the indexed image is searched for; and

calculate a cost function for each motion vector candidate using the indexed image and the indexed reference image.

18. An apparatus according to claim 17, **characterized** in that the pixels included in the part form a square.

19. An apparatus according to claim 18, **characterized** in that the part includes 4 x 4 pixels.

20. An apparatus according to any one of preceding claims 17 to 19, **characterized** in that the apparatus is configured to use a SAD function (Sum of Absolute Differences) as the cost function.

21. An apparatus according to any one of preceding claims 17 to 20, **characterized** in that the parts located close to each other and referred to with indexes partly overlap.

22. An apparatus according to claim 21, **characterized** in that when numbers are calculated for two adjacent parts referred to with indexes, the apparatus is configured to utilize the number calculated for the second part in the calculation of a number for the first part.

23. An apparatus according to any one of preceding claims 17 to 22, **characterized** in that after the motion vector candidate giving the lowest cost function value has been found with an accuracy of one pixel, the apparatus is configured to search for the best motion vector candidate with an accuracy of half a pixel around the motion vector candidate in question.

24. An apparatus according to claim 23, **characterized** in that the apparatus is configured to search the best motion vector candidate with an accuracy of half a pixel as follows:

interpolating values of half a pixel for the indexed candidate block found, which corresponds to the one-pixel motion vector candidate in the reference image, and around the block;

calculating a cost function for each motion vector candidate of half a pixel using the indexed image and interpolated and indexed candidate block:

encoding the image block to be encoded using the motion vector candidate of half a pixel that gives the lowest cost function value.

25. A computer program on a carrier for encoding successive images and comprising computer executable instructions for causing a computer to:

encode an image block to be encoded using the motion vector candidate that gives the lowest cost function value, the motion vector candidate

defining the motion between the image block to be encoded and a candidate block in the search area of a reference image;

characterized by the computer program further comprising computer executable instructions for causing a computer prior to encoding to:

process the image into an indexed image and the reference image into an indexed reference image so that the image and the reference image are divided into parts referred to with indexes and a number is formed from the values of pixels in each part to describe pixel values in a given part;

define a search area in the reference image where the block to be to be encoded in the indexed image is searched for; and

calculate a cost function for each motion vector candidate using the indexed image and the indexed reference image.

26. A computer program according to claim 25, **characterized** in that the pixels included in the part form a square.

27. A computer program according to claim 26, **characterized** in that the part includes 4 x 4 pixels.

28. A computer program according to any one of the preceding claims 25-27, **characterized** in that a SAD function (Sum of Absolute Differences) is used as the cost function.

29. A computer program according to any one of the preceding claims 25-28, **characterized** in that parts located close to each other and referred to with indexes partly overlap.

30. A computer program according to claim 29, **characterized** in that when numbers are calculated for two adjacent parts referred to with indexes, the number calculated for the second part is utilized in the calculation of the number for the first part.

31. A computer program according to any one of the preceding claims 25-30, **characterized** in that after the motion vector candidate giving the lowest cost function value has been found with an accuracy of one pixel, the best motion vector candidate is searched for with an accuracy of half a pixel around the motion vector candidate concerned.

32. A computer program according to claim 31, **characterized** in that the computer program further comprises computer executable instructions for causing the computer to search for the best motion vector candidate with an accuracy of half a pixel as follows:

interpolate values of half a pixel for the indexed candidate block found, which corresponds to the one-pixel motion vector candidate in the reference image, and around the block;

calculate a cost function for each motion vector candidate of half a pixel using the indexed image and interpolated and indexed candidate block; and

encode the image block to be encoded using the motion vector candidate of half a pixel that gives the lowest cost function value.

33. A computer program according to any one of the preceding claims 25-32, **characterized** in that the carrier is at least one of a distribution package, a computer memory and a telecommunications signal.

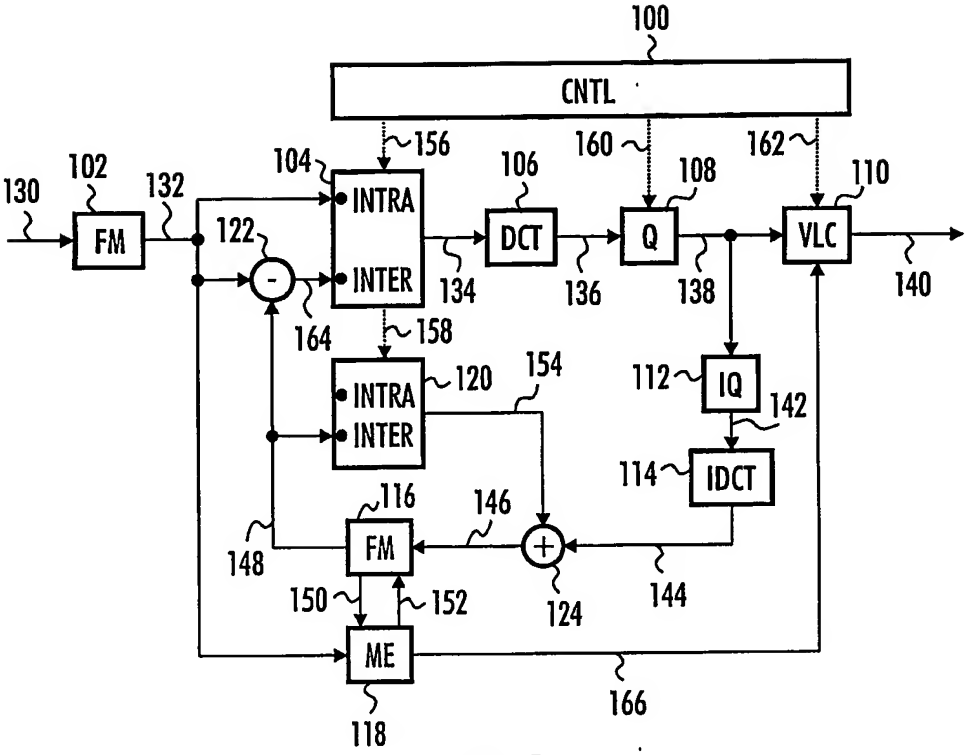

100
CNTL
156
160
162
102
104
106
108
110
130
132
FM
INTRA
INTER
DCT
Q
VLC
122
134
136
138
140
164
158
INTRA
INTER
120
154
112
IQ
142
114
IDCT
116
146
148
FM
150
152
124
144
ME
166
118

FIG. 1

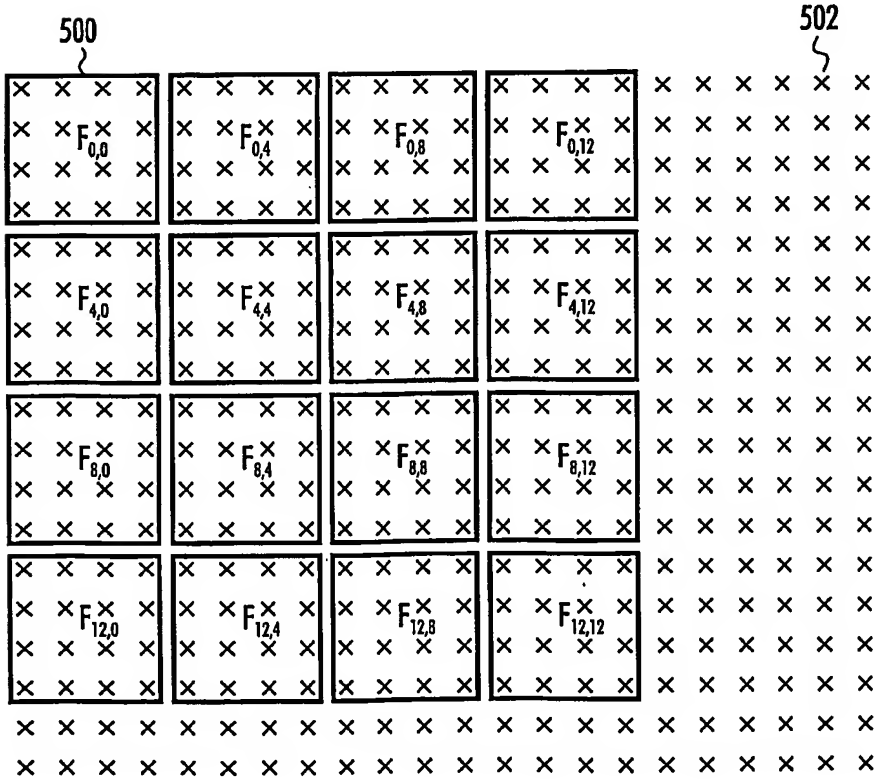

500
502
F0,0
F0,4
F0,8
F0,12
F4,0
F4,4
F4,8
F4,12
F8,0
F8,4
F8,8
F8,12
F12,0
F12,4
F12,8
F12,12

FIG. 5

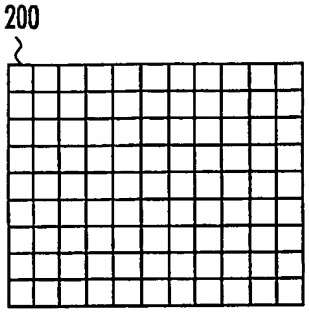


FIG. 2

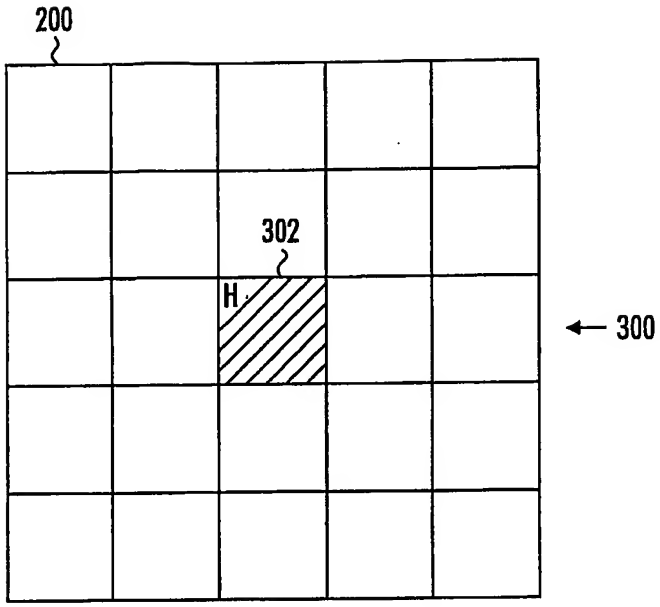


FIG. 3

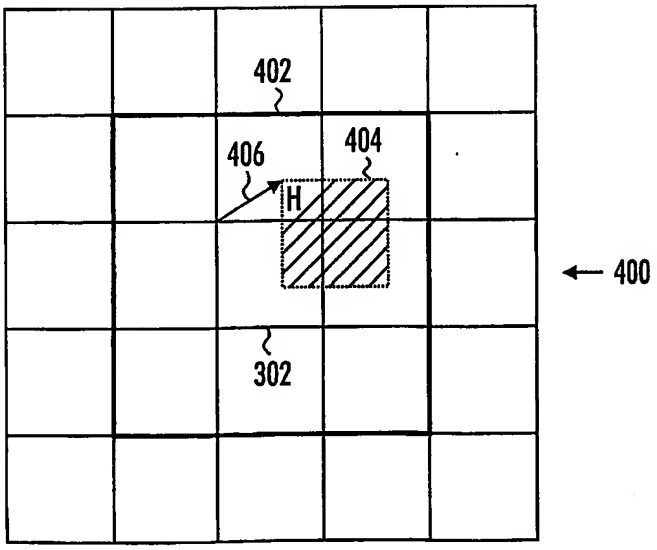


FIG. 4

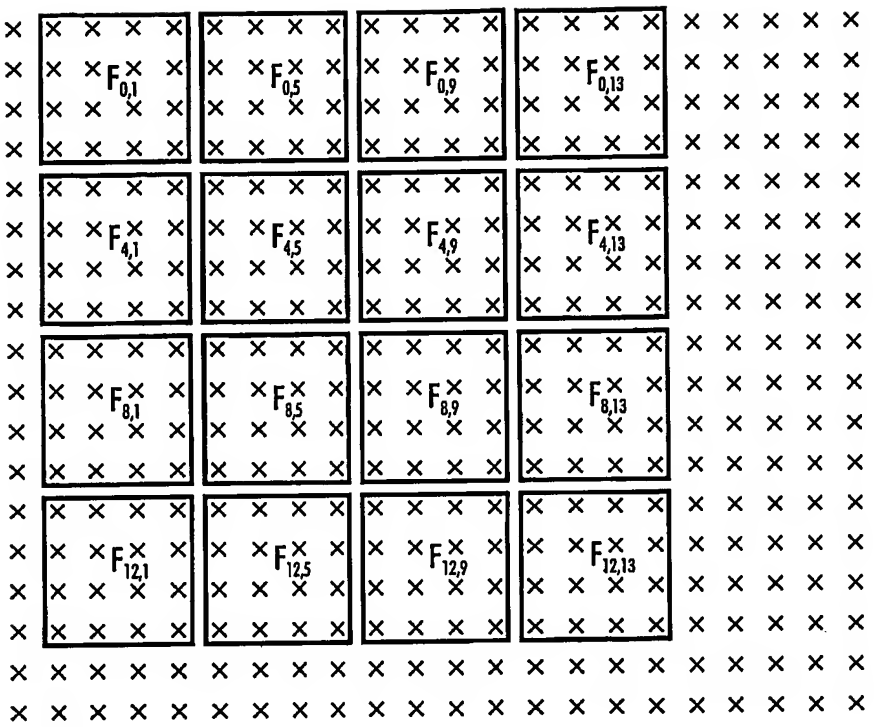

F0,1
F0,5
F0,9
F0,13
F4,1
F4,5
F4,9
F4,13
F8,1
F8,5
F8,9
F8,13
F12,1
F12,5
F12,9
F12,13

FIG. 6

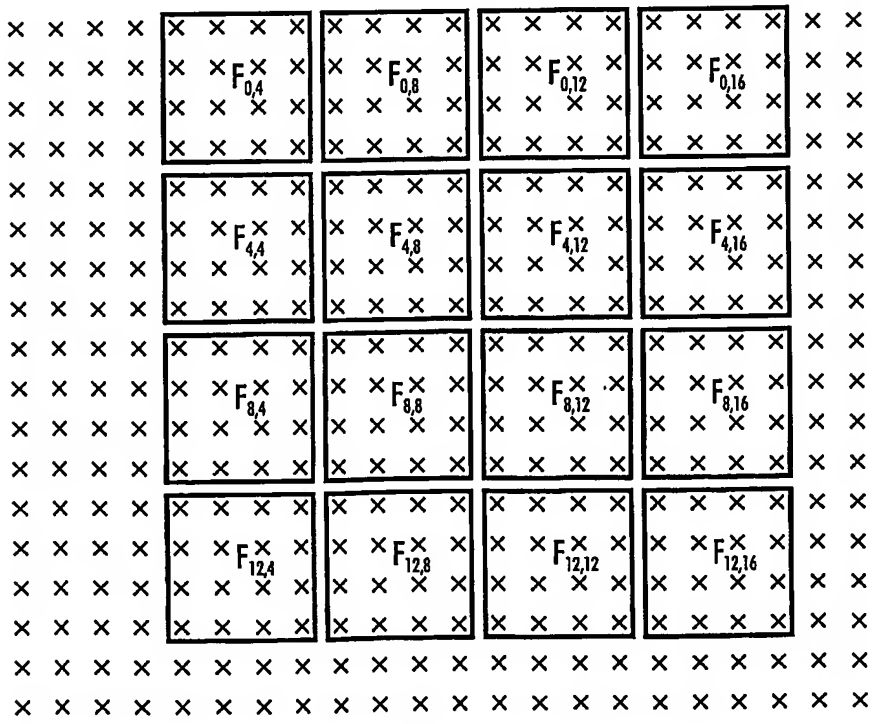

F0,4
F0,8
F0,12
F0,16
F4,4
F4,8
F4,12
F4,16
F8,4
F8,8
F8,12
F8,16
F12,4
F12,8
F12,12
F12,16

FIG. 7

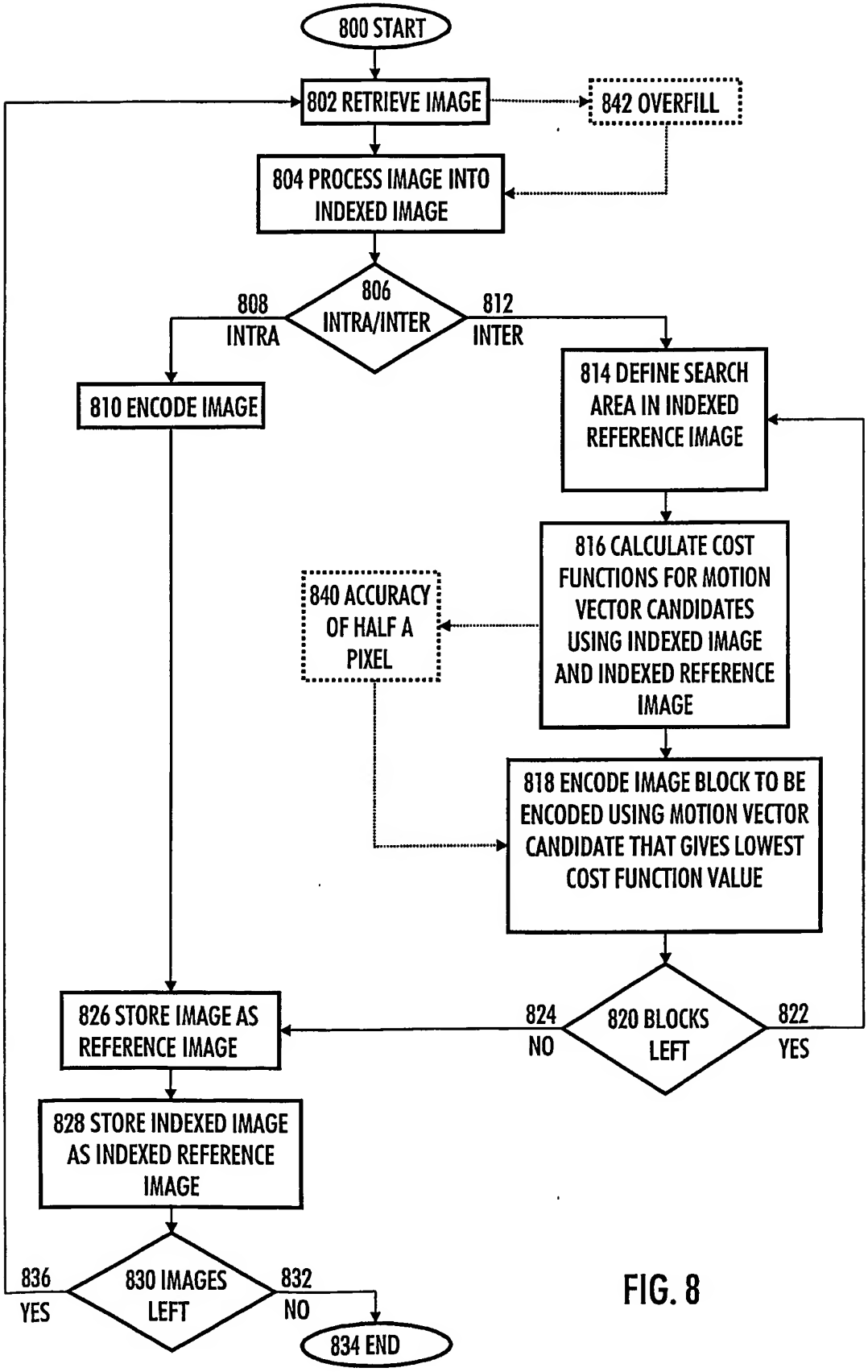

800 START
802 RETRIEVE IMAGE
842 OVERFILL
804 PROCESS IMAGE INTO INDEXED IMAGE
806 INTRA/INTER
808 INTRA
812 INTER
810 ENCODE IMAGE
814 DEFINE SEARCH AREA IN INDEXED REFERENCE IMAGE
816 CALCULATE COST FUNCTIONS FOR MOTION VECTOR CANDIDATES USING INDEXED IMAGE AND INDEXED REFERENCE IMAGE
840 ACCURACY OF HALF A PIXEL
818 ENCODE IMAGE BLOCK TO BE ENCODED USING MOTION VECTOR CANDIDATE THAT GIVES LOWEST COST FUNCTION VALUE
820 BLOCKS LEFT
824 NO
822 YES
826 STORE IMAGE AS REFERENCE IMAGE
828 STORE INDEXED IMAGE AS INDEXED REFERENCE IMAGE
830 IMAGES LEFT
836 YES
832 NO
834 END

FIG. 8

INTERNATIONAL SEARCH REPORT

International application No.
PCT/FI 02/00894

A. CLASSIFICATION OF SUBJECT MATTER

IPC7: H04N 7/26, H04N 7/36, H04N 7/50
According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

IPC7: H04N

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

SE,DK,FI,NO classes as above

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

EPO-INTERNAL, WPI DATA

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	US 5742710 A (HSU, S.C. ET AL), 21 April 1998 (21.04.98), column 2, line 5 - line 39; column 3, line 16 - column 6, line 49, claims 1,3,7, abstract	1-5,7-13, 15-21,23-29, 31-33
A	claim 16	6,14,22,30
	--	
A	US 5610658 A (UCHIDA, M. ET AL), 11 March 1997 (11.03.97), column 7, line 44 - line 50; column 15, line 67 - column 16, line 14, figures 5a,5b,12a, abstract	1-33
	--	
A	EP 0979011 A1 (STMICROELECTRONICS S.R.I.), 9 February 2000 (09.02.00), claim 1, abstract	1-33
	--	

[X] Further documents are listed in the continuation of Box C. [X] See patent family annex.

* Special categories of cited documents:
"A" document defining the general state of the art which is not considered to be of particular relevance
"E" earlier application or patent but published on or after the international filing date
"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)
"O" document referring to an oral disclosure, use, exhibition or other means
"P" document published prior to the international filing date but later than the priority date claimed
"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention
"X" document of particular relevance: the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone
"Y" document of particular relevance: the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art
"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
31 January 2003	05 -02- 2003
Name and mailing address of the ISA/ Swedish Patent Office Box 5055, S-102 42 STOCKHOLM Facsimile No. +46 8 666 02 86	Authorized officer Jesper Bergstrand/LR Telephone No. +46 8 782 25 00

Form PCT/ISA/210 (second sheet) (July 1998)

INTERNATIONAL SEARCH REPORT

International application No.
PCT/FI 02/00894

C (Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	EP 1091592 A2 (MATSUSHITA ELECTRIC IND CO, LTD), 11 April 2001 (11.04.01) --	1-33
A	US 6014181 A (SUN. K.), 11 January 2000 (11.01.00) --	1-33
A	US 6011870 A (JENG, F.-C. ET AL), 4 January 2000 (04.01.00) --	1-33
A	US 5987180 A (REITMEIER, G.A.), 16 November 1999 (16.11.99) --	1-33
A	WO 9941912 A2 (PHILIPS AB), 19 August 1999 (19.08.99) --	1-33

INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No.
PCT/FI 02/00894

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
US 5742710 A	21/04/98	CN 1125375 A	26/06/96
		JP 7262381 A	13/10/95
US 5610658 A	11/03/97	JP 7264603 A	13/10/95
EP 0979011 A1	09/02/00	JP 2000083257 A	21/03/00
		US 6480543 B	12/11/02
EP 1091592 A2	11/04/01	JP 2001112000 A	20/04/01
US 6014181 A	11/01/00	JP 11215503 A	06/08/99
US 6011870 A	04/01/00	NONE	
US 5987180 A	16/11/99	EP 1025537 A	09/08/00
		JP 2001517879 T	09/10/01
		WO 9916011 A	01/04/99
		AU 9585498 A	12/04/99
		AU 9588198 A	12/04/99
		AU 9588298 A	12/04/99
		AU 9670698 A	12/04/99
		AU 9778898 A	12/04/99
		CN 1299562 T	13/06/01
		CN 1302506 T	04/07/01
		EP 1025692 A	09/08/00
		EP 1025697 A	09/08/00
		EP 1025709 A	09/08/00
		EP 1055325 A	29/11/00
		JP 2001517904 T	09/10/01
		JP 2001517906 T	09/10/01
		JP 2002517109 T	11/06/02
		US 5933195 A	03/08/99
		US 6057889 A	02/05/00
		US 6118486 A	12/09/00
		US 6118498 A	12/09/00
		US 6122400 A	19/09/00
		US 2002176506 A	28/11/02
		WO 9916012 A	01/04/99
		WO 9916235 A	01/04/99
		WO 9916242 A	01/04/99
		WO 9916243 A	01/04/99
		WO 9916247 A	01/04/99
		WO 9916253 A	01/04/99
WO 9941912 A2	19/08/99	CN 1256049 T	07/06/00
		DE 69901525 D	00/00/00
		EP 0976251 A,B	02/02/00
		JP 2001520846 T	30/10/01
		US 2002085630 A	04/07/02